

### **REMARKS**

The Examiner rejected claims 1-8, 10-13 and 15-18 under 35 U.S.C. § 103(a) as allegedly being unpatentable over the rules of Algebra.

Applicant respectfully traverses the § 103 rejections with the following arguments.

**35 U.S.C. § 103(a)**

The Examiner rejected claims 1-8, 10-13 and 15-18 under 35 U.S.C. § 103(a) as allegedly being unpatentable over the rules of Algebra.

For example, claims 1, 6, and 7 are not unpatentable over the rules of algebra, because the rules of algebra do not teach or suggest the feature: “compiling said source code into object code, wherein said source code comprises said two algebraic expressions, and wherein said compiling comprises said recasting, said reducing, and said comparing”.

On page 2 of the present office action mailed 10/20/2006, the Examiner argues with respect to a prior “new matter” rejection: “This action is in response to the amendment filed after the Final Office action dated on 07/25/06. The Applicants' filing reply is to result from a request for Interview, on 08/30/06... As in the arguments in the interview, and in the remarks, Applicants point out that compiler compiles source code into executable code has been known and understood in the field of programming, therefore with the new added subject in the abstract can not be new subject matter... Accordingly, Examiner will **withdraw the objection**. However, it would indicate that the features claimed as **compiling said source code into object code**, wherein said source code comprises said two algebraic expressions, and wherein said **compiling comprises** said recasting said reducing, and said comparing., are the Applicants' admitted act, as done by a prior art.”

In response, Applicant acknowledges having admitted, as the Examiner indicates, “that compiler compiles source code into executable code has been known and understood in the field of programming”. However, the preceding admission by Applicant is not an admission of “compiling said source code into object code, **wherein said source code comprises said two**

**algebraic expressions, and wherein said compiling comprises said recasting, said reducing, and said comparing”** (emphasis added) as the Examiner alleges. Applicant has not admitted that compiling source code into object code through use of the claimed steps of recasting, reducing, and comparing is known in the art. Applicant asserts that compiling source code into object code through use of the claimed steps of recasting, reducing, and comparing relates to the novelty and unobviousness of Applicant’s invention

On page 4 of the present office action mailed 10/20/2006, the Examiner argues that “Official notice is also taken that compilation definition in general is to convert source code into object code because it is known in the art, and admitted by Applicants as known and is prior art.... Therefore, it would be obvious to an ordinary in the art to apply rules/notation of algebra to implement the claim, and it would be also obvious to an ordinary in the art to use compilation because it is common in the art, as Applicants already admitted.”

In response, Applicant acknowledges that compiling source code into object code in known in the art. However, Applicant asserts that the rules/notation of algebra do not teach or suggest the feature of compiling source code into object code through use of the claimed steps of recasting, reducing, and comparing. Applicant requests that the Examiner provide a prior art reference allegedly disclosing that the laws of algebra teach or suggest the feature of compiling source code into object code through use of the claimed steps of recasting, reducing, and comparing.

Moreover, Applicant asserts that it is not known in the art to compile source code into object code through use of the claimed steps of recasting, reducing, and comparing. Applicant

notes that the Examiner has not identified any prior art that allegedly teaches or suggests compiling source code into object code through use of the claimed steps of recasting, reducing, and comparing. In the absence of such prior art, the Examiner cannot persuasively establish that it is obvious to compile source code into object code through use of the claimed steps of recasting, reducing, and comparing.

As to the recasting step (“recasting said expressions into a form of one or more token pairs arranged sequentially in a string, each said token pair comprising an operator followed by an operand”), the Examiner argues: “Algebra rules show (a): For example, take  $(a+b)(a-b)$ , one expression and  $a^2-b^2$ , another expression, they are equivalent and will be recasting into a form  $a*a-a*b+a*b-b*b$ , by using the known rules of algebra”.... Therefore, it would be obvious to an ordinary in the art to apply rules/notation of algebra to implement the claim”.

In response, Applicants assert that the Examiner has not identified a prior art reference that allegedly teaches or suggests compiling source code into object code through use of the recasting step.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claims 1, 6, and 7.

As to the reducing step (“reducing said strings in accordance with a set of predetermined simplifying rules”), the Examiner argues: “Algebra rules show (b): For example  $a*a-a*b+a*b-b*b$  is reduced by algebraic rules as  $a*a-b*b$ . (b) reducing said strings in accordance with a set of predetermined simplifying rules”.... Therefore, it would be obvious to an ordinary

in the art to apply rules/notation of algebra to implement the claim”.

In response, Applicants assert that the Examiner has not identified a prior art reference that allegedly teaches or suggests compiling source code into object code through use of the reducing step.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claims 1, 6, and 7.

As to the comparing step (“comparing the reduced strings by matching, to detect equivalence of the two algebraic expressions”), the Examiner argues: “Algebra rules show (c): For example  $(a+b)(a-b) = a*a-b*b$ ; and  $a^2-b^2$  is another expression of  $a*a-b*b$ . In fact  $(a+b)(a-b)$  equals to  $a^2-b^2$ , equals to  $a*a -a*b+a*b-b*b$ . (c) comparing the reduced strings by matching, to detect equivalence of the two algebraic expressions..... Therefore, it would be obvious to an ordinary in the art to apply rules/notation of algebra to implement the claim”.

In response, Applicants assert that the Examiner has not identified a prior art reference that allegedly teaches or suggests compiling source code into object code through use of the comparing step.

In addition, the Examiner has used circular reasoning by arguing that it is obvious to compare the reduced strings by matching to detect equivalence of the two algebraic expressions, in order to detect equivalence of the two algebraic expressions.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claims 1, 6, and 7.

Another reason why claims 1, 6, and 7 are not unpatentable over the rules of algebra is that the Examiner has not provided a persuasive argument as to why it is allegedly obvious to combine steps (a), (b), and (c) sequentially as claimed.

The Examiner has not even addressed the preceding feature of sequential performance of the steps (a), (b), and (c) sequentially as claimed.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claims 1, 6, and 7.

Based on the preceding arguments, Applicants respectfully maintain that claims 1, 6, and 7 are not unpatentable over the rules of Algebra, and that claims 1, 6, and 7 are in condition for allowance. Since claims 2-5 depend from claim 1, Applicants contend that claims 2-5 are likewise in condition for allowance. Since claims 10-13 depend from claim 6, Applicants contend that claims 10-13 are likewise in condition for allowance. Since claims 8 and 15-18 depend from claim 7, Applicants contend that claims 8 and 15-18 are likewise in condition for allowance.

#### Additional Argument For Claim 2

In addition with respect to claim 2, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein the recasting step (a) is preceded by a preconditioning step comprising, in relation to said algebraic expressions, the following sub-steps according to whether a sub step applies:

(da) deleting a space in the expression;

- (db) removing a bracket in the expression by expanding a bracketed sub-expression;
- (dc) inserting a unitary operator at the start of the expression;
- (dd) recasting a power factor, being a variable being raised to a power in the expression, in an alternate form as one of:
  - (dda) the power factor being expressed as the variable multiplied by itself as many times as the power, if the power is a positive integer;
  - (ddb) the power factor being expressed as a reciprocal of the variable multiplied by itself as many times as an absolute value of the power, if the power is a negative integer;
  - (ddc) the power factor being replaced by an appropriate function which can compute the power factor, if the power is not an integer;
- (de) recasting a constant in the expression in exponential format;
- (df) substituting a “+” operator in the expression by “+1\*”, a “1” being in exponential format;
- (dg) substituting a “-” operator in the expression by “-1\*”, a “1” being in exponential format; and
- (dh) recasting a “division by a constant” in the expression as multiplication by a reciprocal of the constant, “wherein said compiling comprises said preconditioning step ”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling comprises the preconditioning subject to the recited limitations of the preconditioning step in claim 2.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 2.

### Additional Argument For Claim 3

In addition with respect to claim 3, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein the simplifying rules in step (b) comprise:

- (ba) arranging token pairs into subgroups;
- (bb) arranging operand tokens in an arranged subgroup in order;
- (bc) reducing the ordered operands by consolidating one or more constants and eliminating variables of opposite effect to form reduced subgroups; and
- (bd) consolidating one or more multiple instances of similar subgroups, to produce a reduced string”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling comprises said reducing said strings in accordance with a set of predetermined simplifying rules, wherein the simplifying rules are subject to the recited limitations in claim 3.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 3.

### Additional Argument For Claim 4

In addition with respect to claim 4, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains an aliased variable, said method further comprising the steps of: arranging an ordered list of aliases of the variable, and substituting a first alias in the



ordered list for all instances of the aliased variable in the expression, wherein said compiling comprises said arranging an ordered list of aliases of the variable and said substituting a first alias in the ordered list”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling comprises said arranging an ordered list of aliases of the variable and said substituting a first alias in the ordered list.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 4.

#### Additional Argument For Claim 5

In addition with respect to claim 5, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains a function, said method further comprising the steps of:

reducing function arguments using the set of predetermined simplifying rules; and replacing the function by a tagged string, said string designating a function name, parameter types, and arguments, wherein the tag distinguishes the function name from a variable, wherein said compiling comprises said reducing function arguments and said replacing the function by a tagged string”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling comprises said reducing function arguments and said replacing the function by a tagged string.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima*

*facie* case of obviousness in relation to claim 5.

#### Additional Argument For Claim 8

In addition with respect to claim 8, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein said reduction process steps comprise:

processing token pairs into an ordered arrangement;

determining that a redundant equivalent subexpression exists within said ordered arrangement; and

responsive to said determining that said redundant equivalent subexpression exists within said ordered arrangement, eliminating said redundant equivalent subexpression from said ordered arrangement”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling process steps for compiling comprise said reduction process steps for reducing, subject to said reduction process steps comprising said processing token pairs, said determining that a redundant equivalent subexpression exists, and said eliminating said redundant equivalent subexpression.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 8.

#### Additional Argument For Claim 10

In addition with respect to claim 10, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein the simplifying rules in step (b)

comprise:

- (ba) arranging token pairs into subgroups;
- (bb) arranging operand tokens in an arranged subgroup in order;
- (bc) reducing the ordered operands by consolidating one or more constants and eliminating variables of opposite effect to form reduced subgroups; and
- (bd) consolidating one or more multiple instances of similar subgroups, to produce a reduced string”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said means for compiling comprises said reduction means for reducing said strings in accordance with a set of predetermined simplifying rules, wherein the simplifying rules are subject to the recited limitations in claim 10.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 10.

#### Additional Argument For Claim 11

In addition with respect to claim 11, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains an aliased variable, said apparatus further comprising:

means for arranging an ordered list of aliases of the variable; and

means for substituting a first alias in the ordered list for all instances of the aliased variable in the expression, wherein said means for compiling comprises said means for arranging an ordered list of aliases and said means for substituting a first alias in the ordered list”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said means for compiling comprises said means for arranging an ordered list of aliases and said means for substituting a first alias in the ordered list.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 11.

#### Additional Argument For Claim 12

In addition with respect to claim 12, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains a function, said apparatus further comprising:

means for reducing function arguments using the set of predetermined simplifying rules;  
and

means for replacing the function by a tagged string, said string designating a function name, parameter types, and arguments, wherein the tag distinguishes the function name from a variable, wherein said means for compiling comprises said means for reducing function arguments and said means for replacing the function”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said means for compiling comprises said means for reducing function arguments and said means for replacing the function.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 12.

### Additional Argument For Claim 13

In addition with respect to claim 13, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein said reduction means for reducing said strings comprises:

means for processing token pairs into an ordered arrangement;

means for determining whether a redundant equivalent subexpression exists within said ordered arrangement; and

means for eliminating said redundant equivalent subexpression from said ordered arrangement, wherein said means for compiling comprises said means for processing token pairs, said means for determining whether a redundant equivalent subexpression exists, and said means for eliminating said redundant equivalent subexpression”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said means for compiling comprises said means for processing token pairs, said means for determining whether a redundant equivalent subexpression exists, and said means for eliminating said redundant equivalent subexpression.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 13.

### Additional Argument For Claim 15

In addition with respect to claim 15, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein the simplifying rules in step (b) comprise:

- (ba) arranging token pairs into subgroups;
- (bb) arranging operand tokens in an arranged subgroup in order;
- (bc) reducing the ordered operands by consolidating one or more constants and eliminating variables of opposite effect to form reduced subgroups; and
- (bd) consolidating one or more multiple instances of similar subgroups, to produce a reduced string”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling process steps for compiling comprise said reduction process steps for reducing said strings in accordance with a set of predetermined simplifying rules, wherein the simplifying rules are subject to the recited limitations in claim 15.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 15.

#### Additional Argument For Claim 16

In addition with respect to claim 16, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains an aliased variable, said computer program further comprising the steps of:

arranging steps for arranging an ordered list of aliases of the variable, and substituting steps for substituting a first alias in the ordered list for all instances of the aliased variable in the expression, wherein said compiling process steps for compiling comprise said arranging steps for arranging an ordered list of aliases”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling comprises process steps for compiling comprise said arranging steps for arranging an ordered list of aliases.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 16.

#### Additional Argument For Claim 17

In addition with respect to claim 17, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein an algebraic expression whose equivalence is to be determined contains a function, said computer program further comprising the steps of:

reducing steps for reducing function arguments using the set of predetermined simplifying rules; and

replacing steps for replacing the function by a tagged string, said string designating a function name, parameter types, and arguments, wherein the tag distinguishes the function name from a variable, wherein said compiling process steps for compiling comprise said reducing steps for reducing function arguments and said replacing steps for replacing the function by a tagged string”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling process steps for compiling comprise said reducing steps for reducing function arguments and said replacing steps for replacing the function by a tagged string.

Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 17.

#### Additional Argument For Claim 18

In addition with respect to claim 18, Applicants respectfully maintain that the rules of algebra do not teach or suggest the feature of: “wherein said reduction process steps comprise:

processing steps for processing token pairs into an ordered arrangement;

determining steps for determining whether a redundant equivalent subexpression exists within said ordered arrangement; and

eliminating steps for eliminating said redundant equivalent subexpression from said ordered arrangement”.

Applicant respectfully asserts that the Examiner has not identified any prior art that allegedly discloses that said compiling process steps for compiling comprise said reduction process steps for reducing which comprise said processing steps for processing token pairs, said determining steps for determining whether a redundant equivalent subexpression exists, and eliminating steps for eliminating said redundant equivalent subexpression.

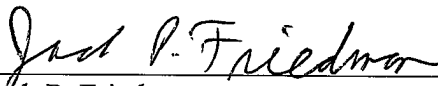
Therefore, Applicants respectfully contend that the Examiner has not established a *prima facie* case of obviousness in relation to claim 18.



## CONCLUSION

Based on the preceding arguments, Applicant respectfully believes that all pending claims and the entire application meet the acceptance criteria for allowance and therefore request favorable action. If the Examiner believes that anything further would be helpful to place the application in better condition for allowance, Applicant invites the Examiner to contact Applicant's representative at the telephone number listed below. The Director is hereby authorized to charge and/or credit Deposit Account 09-0457.

Date: 01/22/2007

  
\_\_\_\_\_  
Jack P. Friedman  
Registration No. 44,688

Schmeiser, Olsen & Watts  
22 Century Hill Drive - Suite 302  
Latham, New York 12110  
(518) 220-1850